AMSnet 2.0: A Large AMS Database with AI Segmentation for Net Detection

Yichen Shi*§, Zhuofu Tao*†, Yuhao Gao§, Li Huang§, Hongyang Wang§
Zhiping Yu¶, Ting-Jung Lin§‖, Lei He §‖
§Ningbo Institute of Digital Twin, Eastern Institute of Technology, Ningbo, China
†University of California, Los Angeles, USA, ¶Tsinghua University, Beijing, China
‖tlin@idt.eitech.edu.cn, lei.hexun@hotmail.com

***Abstract*—Multimodal large language models (MLLM) struggle to understand circuit schematics due to their limited recognition capabilities. This could be attributed to the lack of high-quality schematic-netlist training data. Existing work such as AMSnet applies schematic parsing to generate netlists. However, these methods rely on hard-coded heuristics and are difficult to apply to complex or noisy schematics in this paper. We therefore propose a novel net detection mechanism based on segmentation with high robustness. The proposed method also recovers positional information, allowing digital reconstruction of schematics. We then expand the AMSnet dataset with schematic images from various sources and create AMSnet 2.0. AMSnet 2.0 contains 2,686 circuits with schematic images, Spectre-formatted netlists, OpenAccess digital schematics, and positional information for circuit components and nets, whereas AMSnet only includes 792 circuits with SPICE netlists but no digital schematics.**

***Index Terms*—AMS circuit design, MLLM, circuit topology, front-end design**

I. Introduction

Researchers employ multimodal large language models (MLLMs) in various applications in analog and mixed-signal (AMS) circuit design, such as topology design [1]–[4], sizing [5], layout generation [6], design rule check (DRC) code generation [7], and so on. This demonstrates that these MLLMs possess abundant field knowledge. However, current MLLMs still face difficulties in schematic recognition, understanding [8], and netlist generation. Generated data suffer from incorrectness due to hallucination, and generally cannot be used for their intended purposes, such as simulation.

One of the main reasons for the limited ability of MLLMs to recognize circuit schematics is the lack of high-quality multimodal training data, such as schematic-netlist pairs. Existing datasets either only focus on a single modality, such as image or language, or are too small to use for (M)LLM training. To address this issue and collect more high-quality data, we developed and released a data labeling platform, where the user uploads schematics and labels their schematic elements, wires, and expert insights, as shown in Fig. 1. Backed by this platform, we construct AMSnet 2.0, a large-scale AMS dataset containing schematics, netlists, and position information for schematic elements and nets.

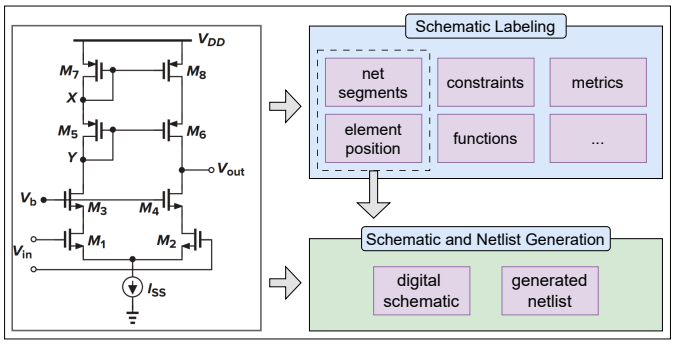

Fig. 1: AMSnet 2.0 workflow for multimodal dataset construction.

In addition to manual annotations of circuit information, establishing the correspondence between schematic images and netlists is crucial for the dataset. However, the process often relies on manually interpreting images and then writing the netlists, which was time-consuming and error-prone. To address this issue, previous work [8], [9] proposes image processing-based methods to generate netlists from schematics to naturally form desired multi-modal pairs. Some methods include template matching [10] to detect schematic elements and the Hough transform [11] to identify wires. These methods are able to establish connectivity between schematic elements and generate netlists. However, they generally require schematic images to be clear and free of additional markings, which presents deficiencies in efficiency and robustness. It is difficult for these methods to process *noisy* images, as shown in Fig. 2.

On the other hand, the above methods process wire pixels into nets via graph search algorithms or image transformations. The problem with these methods is that they only observe wire pixels as is and do not use any *contextual information*, such as where the elements are and what they connect to. As a result, they may have trouble distinguishing wires from other illustrative markings (e.g. the boxes in Fig. 2 (a) can be considered as wires). To compensate, these studies may assume that markings tend to show up in different colors, and then use techniques such as binarization to filter them out in a preprocessing step. However, this could introduce additional

This work was partially supported by "Science and Technology Innovation in Yongjiang 2035" (2024Z283) and by research support from BTD Inc.
*Equal contribution
**Corresponding authors

979-8-3503-7608-1/24$31.00 ©2024 IEEE

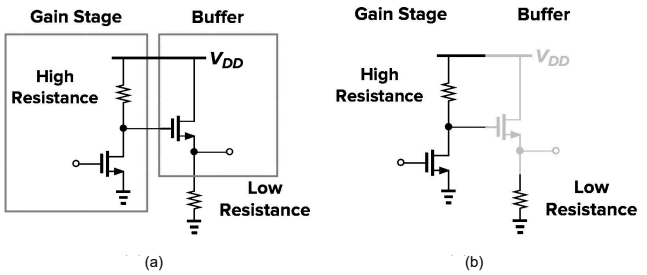

Fig. 2: Examples of noisy schematics: (a) overlaid markings and (b) partial highlighting

issues with a different type of illustration, as shown in Fig. 2 (b), where parts of the schematic may be incorrectly filtered.

This paper presents a more robust method for net detection. It uses a deep-learning-based image instance segmentation model to accurately determine whether lines represent wires based on contextual information. By offloading the decision process to the training dataset, our method effectively differentiates wires from markings and is significantly more robust than hard-coded heuristics. We evaluate it on the AMSnet 2.0 testing set, which we divide into three difficulty levels based on schematic complexity, presence of illustrative markings, and image quality. Using a confusion matrix to analyze element types and connections in the netlist, we achieve F1 scores of 90.19%, 84.17%, and 80.39% for netlist generation across these splits, respectively.

Furthermore, instance segmentation provides us with precise positional information about wires, allowing us to perform skeletonization to extract key coordinates such as ends, diverges, and turns, and summarize nets as sets of line segments. This enables the automatic reconstruction of digital schematics in widely-used formats such as OpenAccess, so that engineers can conveniently access schematics in EDA software, without having to manually modify netlists.

To summarize, our contributions in this work are as follows.

- We construct and release AMSnet 2.0, a large-scale AMS dataset containing image and digital schematics, netlist, and position data for schematic elements and wires.
- We develop a more robust algorithm for schematic parsing and netlist generation, eliminating the need for hard-coded heuristics in net detection.
- We develop a labeling platform for AMS schematics, and release it at this link to support expansion of AMSnet 2.0.

The remainder of this paper is organized as follows. Section II introduces the schematic labeling platform. Section III presents the algorithms for schematic and netlist generation based on circuit images. Section IV shows the experimental results. Finally, Section V discusses potential future work based on AMSnet 2.0 and concludes.

II. Schematic Labeling

The section describes the construction process for AMSnet 2.0, which includes image collection, schematic element and net detection, and netlist generation. We collect 2686 circuit schematics from textbooks [12] and public competitions [13], and manually annotate them. Fig. 1 presents an overview.

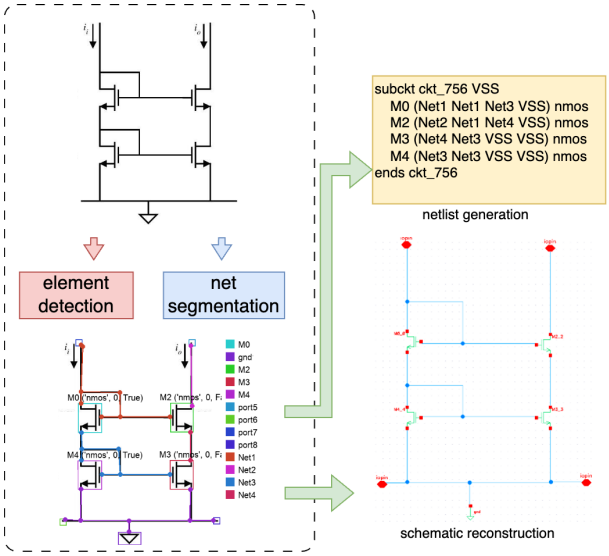

Fig. 3: The full schematic processing pipeline: schematic element detection, net segmentation, Spectre format netlist generation, and digital schematic reconstruction in OpenAcess format.

a) Element Labeling: Users upload circuit schematics and draw bounding boxes for schematic elements on the graphical interface. They then assign the correct category labels (e.g., PMOS). The dataset is in YOLO format, including the coordinates of the object center and its width and height. Following [13], we also annotate the cross-points of all wires, categorizing whether they are visually intersecting or functionally connected.

b) Net Labeling: Users label all the nets in a schematic image. Specifically, a net consists of one or more line segments. The users label each line segment by tracing them on the graphical interface, using a different net instance for each net. This groups all segments of the same net under the same instance, distinguishing them from other nets. The dataset contains each net and the segments it comprises, including the coordinates (start and end points) of each segment.

c) Insight Labeling: Engineers and experts can label the type (e.g. OPAMP), name (e.g. cascode), function (e.g. signal amplification), and characteristics (e.g. high gain) of the circuit topology based on the schematic. We store this data in the form of key-value pairs. These expert insights will play a crucial role in driving the MLLM-driven automatic AMS design in the future.

III. Schematic and Netlist Generation

Fig. 3 presents the proposed methods of netlist and schematic generation. We use a deep-learning-based vision model to recognize the *schematic elements (i.e. voltage sources, capacitors, transistors, amplifier symbols, ground*

symbols, etc.) and nets in the input images. Based on the recognition results, our method automatically reconstructs the digital schematics and generates the corresponding netlists.

A. Device Detection

As in previous work [4], [8], [9], [13] demonstrates, SOTA object detection models are able to effectively identify schematic elements. However, our approach extends beyond previous efforts by also detecting and recognizing the intersections of circuit nets. Specifically, when two wires meet at a cross-point, the presence of a junction (i.e. a dot) indicates that the wires belong to the same net; if no junction is present, we consider them separate nets.

B. Wire Segmentation

Previous work on wire segmentation and net detection often relies on *proposal*-based instance segmentation methods [14], [15]. These methods first generate potential bounding box proposals and then apply segmentation through a dedicated module. However, they struggle to fully encapsulate all pixels of a wire within a single proposal and may include pixels from multiple nets, leading to suboptimal segmentation. Additionally, [16] explores end-to-end instance segmentation techniques, but it still faces challenges in accurately distinguishing intersecting nets.

To address these limitations, we propose a more robust and efficient two-stage approach. The flow starts with a semantic segmentation network (U-net [17]) to segment all wires and produce *wire masks*, as shown in Fig. 4(a). The extracted wire masks are then post-processed to accurately separate intersecting nets, as shown in Fig. 4(b). Our method eliminates the need for bounding box proposals, substantially improving segmentation accuracy and ensuring precise net delineation.

C. Mocked Marking Data Augmentation

Though the aforementioned methods perform well on *clean* schematics, they still struggle in net segmentation for *noisy* schematics. An example is shown in Fig. 2 (a), where rectangles are drawn over the schematics. These *noises* are usually overlay markings for illustration purposes; however, they significantly interfere with schematic interpretation.

Data augmentation is a standard practice in machine learning, where raw training data is altered to improve the robustness of the trained model. In this case, given sufficient training data and a reasonable data distribution, the proposed model is capable of handling data noise natively. We therefore augment raw schematics with rectangles and text markings, without changing the net mask labeling. We randomly generate *mocked* markings over the schematics, and ensure that the placements of the rectangles do not directly cross over the schematic elements. After this augmentation step, the trained model is able to successfully detect wires through the mocked markings and handle these special cases.

D. Net Recognition

After device and wire recognition, the flow needs to understand how they are connected to form nets. Given the segmented wire masks from the above steps, those that do not intersect each other belong to separate connected components. We assign a unique net label to each connected component. Fig. 4 (b) shows an example of wire masks that interconnect and are affected by cross lines. In this case, a mask is applied to the area around the intersection points, which splits the connected component at the intersection. Then, the tool merges the opposite connections to precisely identify the locations of multiple nets. Specifically, it orders all four connections to the intersection based on the angle between the connecting points and the center of the intersection, as shown in the green and purple numbers 1, 2, 3, 4 in Fig. 4(b). Finally, the connections 1, 3 and 2, 4 are grouped to resolve the intersection.

E. Netlist Generation & Schematic Reconstruction

After detecting each schematic element and its rotation angle, we can identify the pin areas for each element (i.e., the pins of a resistor are located at the midpoint of each end of its bounding box). By identifying the nearest or intersecting net labels for each element and its corresponding pins, the tool can establish the connectivity relationship and generate a precise netlist.

So far, the tool collected the coordinates of the schematic elements and net segments to reconstruct editable digital schematics in widely-used formats, such as OpenAccess. This process ensures that the generated electronic schematics are readable without the need for manual circuit construction and allows for further modifications and simulations.

IV. Experiments

A. Experiment Setting

We train the models on labeled data with a training – validation - testing split of 1986 - 500 - 200 schematics. We employ YOLO11 for element detection, training it for 500 epochs with a batch size of 16, along with other default parameters of YOLO11. For wire segmentation, we use two models, UNet and YOLO11-seg, and compare their results. We train UNet for 1000 epochs with a batch size of 32, and YOLO11-seg for 1500 epochs with a batch size of 16, both with default parameters. We conduct all experiments on an NVIDIA RTX 4090 GPU.

B. Evaluation Protocol

For the evaluation of netlist generation, we select three test sets with varying levels of difficulty—easy (80), medium (70), and hard (50)—based on the complexity of the circuit topology as well as the quality and style of the images. Specifically, we manually evaluate the test set schematics on four metrics: high element count ($\geq$ 10), crossing wires, overlaid markings, and low resolution. Images satisfying none of the above are labeled “easy”, images satisfying exactly one of the above are “medium”, and the rest are “hard”. Fig. 5 presents three examples, each from the easy, medium, and hard splits, along

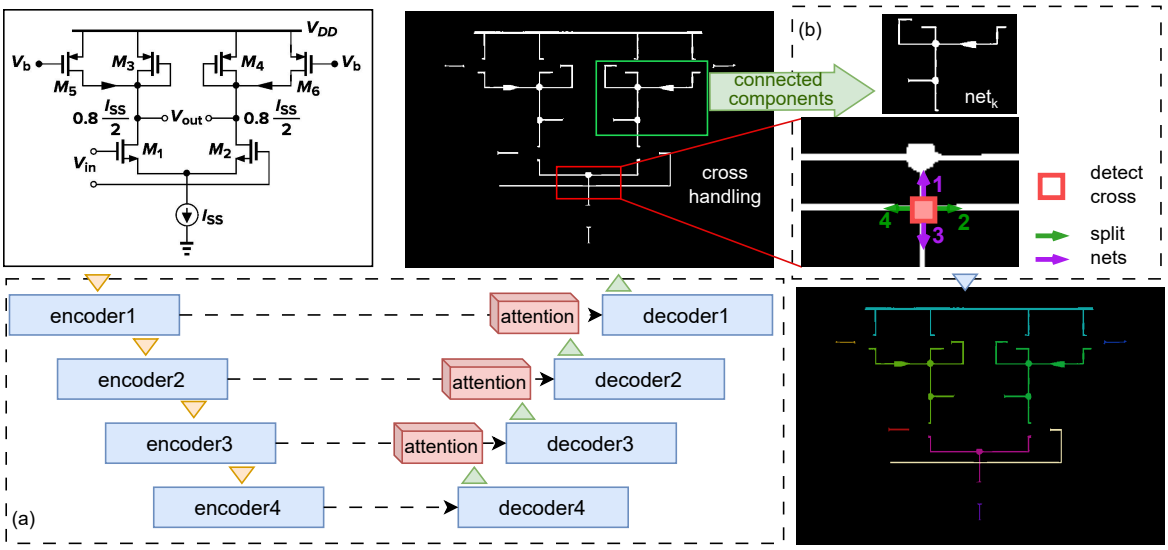

Fig. 4: The proposed two-stage net detection method. (a) Semantic segmentation on the circuit wires to delineate the wires and background areas. (b) Detecting connected components, then perform split and merge operations at intersection points to identify the nets.

Model	Easy	Medium	Hard
Unet [17]	90.19	84.17	80.39
YOLO11-seg [15]	83.62	71.10	73.11

TABLE I: Netlist generation F1 scores

with resulted element detection, net detection, and Spectre netlist generated by the UNet-based procedure.

C. Netlist Generation Results

As previously mentioned, we apply two options for the wire detection in our experiments. One is YOLO11-seg [15], which directly detects wires into separate nets. The other is UNet, which first detects wire chunks, and then algorithmically post-processes them into separate nets with the help of crossing points detected during element detection. We therefore report two sets of results, as shown in Table I.

The reported scores are evaluated via a confusion matrix. For each netlist component (schematic elements excluding VDD, GND, ports, etc.) and net connection *in the ground truth netlist*, if it was correctly predicted in the predicted netlist, we mark this as a true positive. If it was missing or predicted incorrectly, we mark this as a false negative. Finally, if the element or connection does not exist but was predicted (i.e. extra capacitor, or extra connection on a component, etc.), we mark this as a false positive. From this point, we use the standard precision-recall-F1 definition, and report the F1 scores in Table I. Note that incorrect type prediction, such as predicting capacitors as resistors, is also a false negative.

Since different netlists could have different element names and net names, direct comparison may undermine the results; we therefore consider all permutations between ground truth names and predicted names. For example, for netlist ckt_192 shown in Fig. 5 (a), the element names for the predicted netlist include I0 and M9-12, and the net names include VDD, VSS, and net2-6. Suppose the ground truth VDD is labeled "net1", the correct permutation should match net1 against VDD among others. We report the F1 score generated by the best possible permutation. Since the computation complexity for permutations grows exponentially, we manually determine the F1 score when the schematics are too complex.

D. Schematic Reconstruction Results

Fig. 6 presents the schematic reconstruction results for the images shown in Fig. 5. The automatic generation files are in OpenAccess format and displayed in Cadence Virtuoso. Compared with the original images, we can see that our flow has successfully reconstructed the digital schematics These automatically reconstructed schematics accurately reproduce the element layout, net positions, and connections depicted in the images, requiring minimal manual intervention.

V. Conclusions and Discussions

In this work, we introduce AMSnet 2.0, a larger-scale dataset that includes schematics images, Spectre format netlists, and position information. We propose a method based on instance segmentation for net detection. Utilizing the detected elements and net position information, we implement an automatic tool that can generate netlists and reconstruct schematics in OpenAccess format. We release our data annotation platform, allowing the community to annotate circuit data and further enlarge the dataset.

A. AMSnet 2.0 Statistics

Fig. 8 presents the data distribution for the 2686 circuits in AMSnet 2.0. We can see that the majority of circuits include around 10-20 elements and nets, but the larger ones could

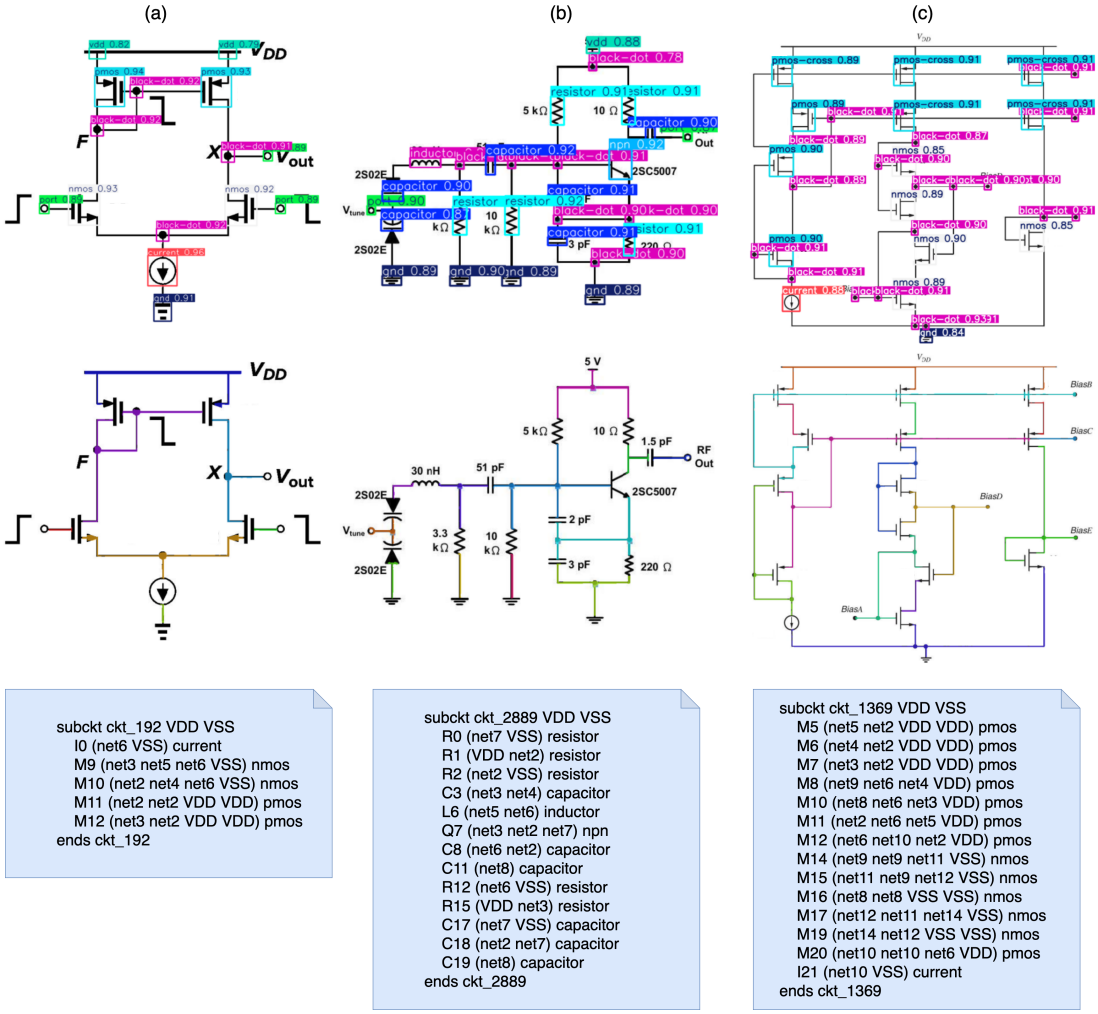

Fig. 5: Results for element detection (top), net detection (middle), and Spectre format netlist generation (bottom), from the UNet procedure. Columns (a), (b), and (c) present examples from easy, medium, and hard splits respectively.

contain up to 80. Fig. 9 presents the element type distribution; we can see that AMSnet 2.0 contains hundreds to thousands of each type. The full dataset will be open-sourced upon the acceptance of this paper, along with corresponding Spectre format netlists, OpenAccess format schematics, and position information for all elements and net segments.

B. Future Applications of AMSnet2.0

The multimodal AMSnet 2.0 establishes the correspondence between schematics and netlists and collects a large amount of human design experience through the annotation platform. With AMSnet 2.0 as the dataset, it is possible to empower MLLMs or other AI models to address key challenges in AMS circuit design in the future.

a) Schematics Generation from Netlist: Converting a netlist into a readable schematic can help designers quickly understand an AMS circuit, which has been a long-standing EDA challenge. With the extensive amount of schematic-netlist pairs in AMSnet 2.0, we plan to train an LLM to facilitate the conversion of netlists to readable schematic diagrams. This work will greatly ease the understanding and debugging of AMS circuits generated by LLMs. It will also enable the creation of more schematic-netlist pairs and further enlarge AMSnet 2.0.

b) LLM Enhanced for AMS Circuit Understanding: Due to the current lack of datasets that include large-scale, high-quality AMS circuit schematics and netlists, the performance of SOTA LLMs and MLLMs in understanding circuit schematics remains suboptimal [8]. Specifically, they struggle to accurately generate the circuit netlists. Building on AMSnet 1.0, the 2.0 dataset includes a richer set of visual information with the introduction of key-point coordinates. This data is crucial for training AMS-specific MLLMs to enhance their understanding of circuit schematics. We conduct supervised fine-tuning (SFT) of the MLLM using the existing data. As shown in Fig. 7, it is evident that after SFT, the MLLM possesses basic capabilities in detecting and locating elements in simple circuit schematics, although their performance on more complex diagrams still needs improvement.

In the future, we expect continuous growth of AMSnet

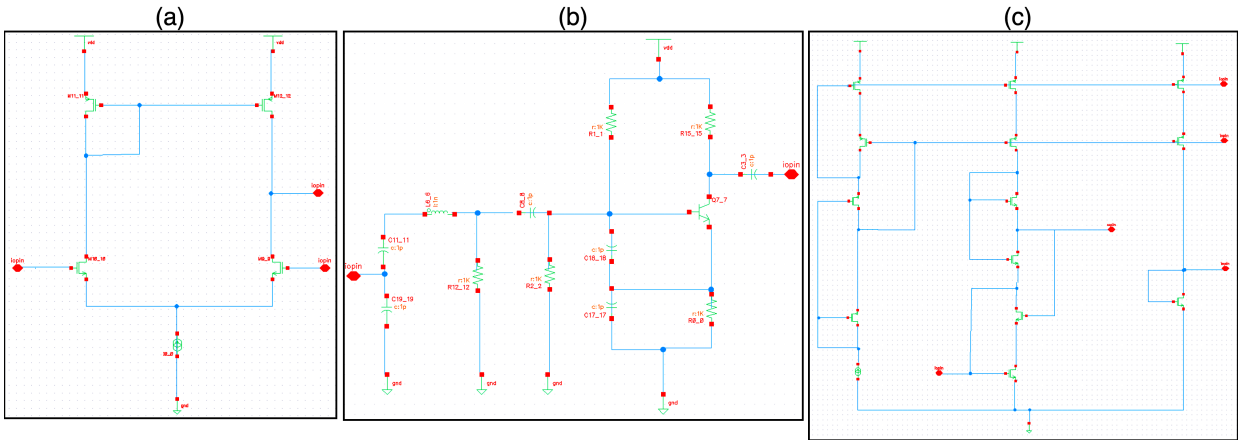

Fig. 6: Generated schematics in OpenAccess format

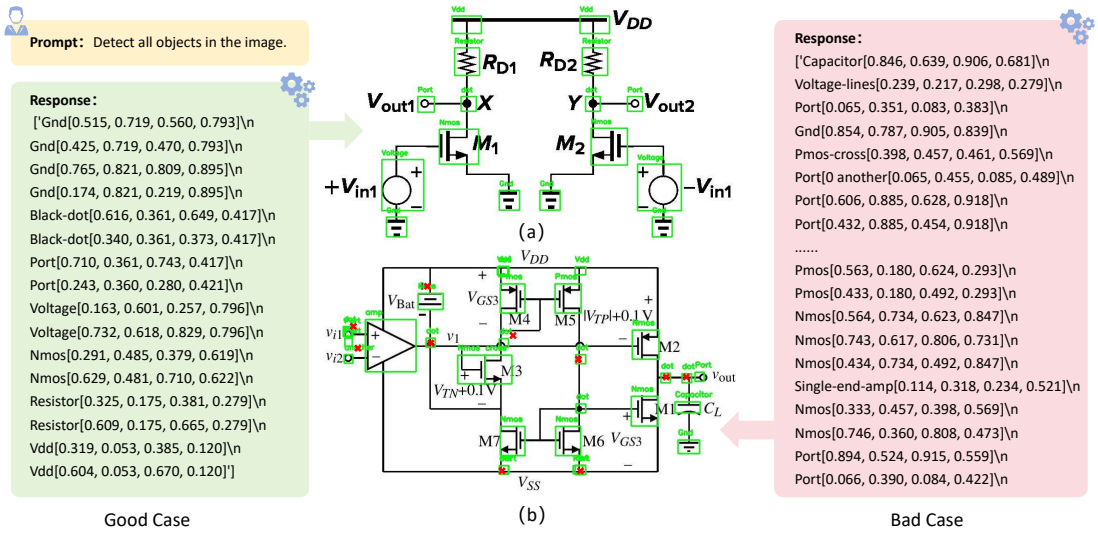

Fig. 7: LLM-based schematic analysis after SFT based on AMSnet 2.0

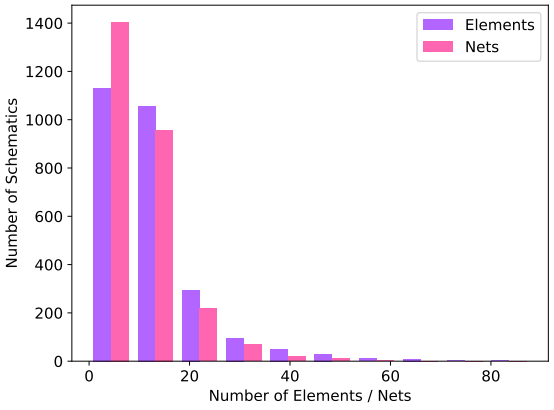

Fig. 8: Distribution of schematic complexity by numbers of elements and mets

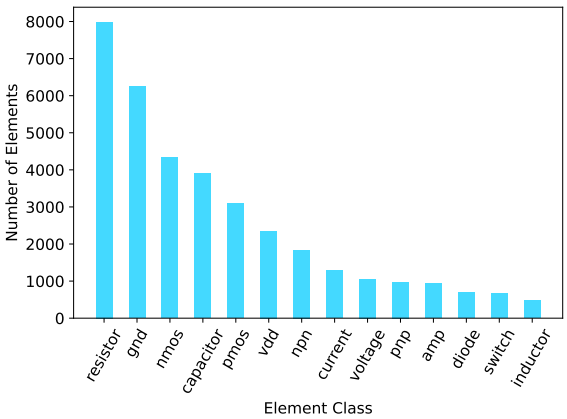

Fig. 9: Distribution of schematic element types

2.0 enabled by the image-to-netlist and netlist-to-schematic pipelines. The dataset will be used for SFT and RL in SOTA MLLMs. Our goal is to facilitate MLLMs with robust capabilities in understanding AMS circuit schematics.

REFERENCES

[1] C.-C. Chang, Y. Shen, S. Fan, J. Li, S. Zhang, N. Cao, Y. Chen, and X. Zhang, "LaMAGIC: Language-Model-Based Topology Generation for Analog Integrated Circuits," *arXiv preprint arXiv:2407.18269*, 2024.

[2] Y. Lai, S. Lee, G. Chen, S. Poddar, M. Hu, D. Z. Pan, and P. Luo, "AnalogCoder: Analog Circuit Design via Training-Free Code Generation," *arXiv preprint arXiv:2405.14918*, 2024.

[3] J. Gao, W. Cao, J. Yang, and X. Zhang, "AnalogGenie: A Generative Engine for Automatic Discovery of Analog Circuit Topologies," in *The 13th International Conference on Learning Representations*, 2025.

[4] Y. Shi, Z. Tao, Y. Gao, T. Zhou, C. Chang, Y. Wang, B. Chen, G. Zhang, A. Liu, Z. Yu, T.-J. Lin, and L. He, "AMSnet-KG: A Netlist Dataset for LLM-based AMS Circuit Auto-Design Using Knowledge Graph RAG," *arXiv preprint arXiv:2411.13560*, 2024.

[5] Y. Yin, Y. Wang, B. Xu, and P. Li, "ADO-LLM: Analog Design Bayesian Optimization with In-Context Learning of Large Language Models," *arXiv preprint arXiv:2406.18770*, 2024.

[6] B. Liu, H. Zhang, X. Gao, Z. Kong, X. Tang, Y. Lin, R. Wang, and R. Huang, "LayoutCopilot: An LLM-powered Multi-agent Collaborative Framework for Interactive Analog Layout Design," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2025.

[7] C.-C. Chang, C.-T. Ho, Y. Li, Y. Chen, and H. Ren, "DRC-Coder: Automated DRC Checker Code Generation Using LLM Autonomous Agent," *arXiv preprint arXiv:2412.05311*, 2024.

[8] J. Bhandari, V. Bhat, Y. He, S. Garg, H. Rahmani, and R. Karri, "Auto-SPICE: Leveraging LLMs for Dataset Creation via Automated SPICE Netlist Extraction from Analog Circuit Diagrams," *arXiv preprint arXiv:2411.14299*, 2024.

[9] Z. Tao, Y. Shi, Y. Huo, R. Ye, Z. Li, L. Huang, C. Wu, N. Bai, Z. Yu, T.-J. Lin, and L. He, "AMSnet: Netlist Dataset for AMS Circuits," in *2024 IEEE LLM Aided Design Workshop (LAD)*, pp. 1–5, IEEE, 2024.

[10] J. P. Lewis *et al.*, "Fast Template Matching," in *Vision interface*, vol. 95, pp. 15–19, Quebec City, QC, Canada, 1995.

[11] J. Illingworth and J. Kittler, "A Survey of the Hough Transform," *Computer vision, graphics, and image processing*, vol. 44, no. 1, pp. 87–116, 1988.

[12] B. Razavi, *Design of Analog CMOS Integrated Circuits*. 2005.

[13] "2024 China Postgraduate IC Innovation Competition · EDA Elite Challenge Contest." http://edachallenge.cn.

[14] D. Bolya, C. Zhou, F. Xiao, and Y. J. Lee, "YOLACT: Real-Time Instance Segmentation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9157–9166, 2019.

[15] G. Jocher and J. Qiu, "Ultralytics YOLO11," 2024.

[16] A. M. Hafiz and G. M. Bhat, "A Survey on Instance Segmentation: State of the Art," *International Journal of Multimedia Information Retrieval*, vol. 9, no. 3, pp. 171–189, 2020.

[17] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional Networks for Biomedical Image Segmentation," in *The 18th International Conference of Medical Image Computing and Computer-assisted Intervention (MICCAI)*, pp. 234–241, Springer, 2015.